physical sciences forum

MDPI

Proceeding Paper

Variational Bayesian Approximation (VBA) with Exponential Families and Covariance Estimation [†]

Seyedeh Azadeh Fallah Mortezanejad [1] and Ali Mohammad-Djafari [2,3,*]

[1] School of Automotive and Traffic Engineering, Jiangsu University, Zhenjiang 212013, China; azadehfallah131@gmail.com
[2] International Science Consulting and Training (ISCT), 91440 Bures sur Yvette, France
[3] Shanfeng Company, Shaoxing 312352, China
[*] Correspondence: djafari@ieee.org

[†] Presented at the 42nd International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering, Garching, Germany, 3–7 July 2023.

Abstract: Variational Bayesian Approximation (VBA) is a fast technique for approximating Bayesian computation. The main idea is to assess the joint posterior distribution of all the unknown variables with a simple expression. Mean–Field Variational Bayesian Approximation (MFVBA) is a particular case developed for large–scale problems where the approximated probability law is separable in all variables. A well–known drawback of MFVBA is that it tends to underestimate the variances in the variables, even though it estimates the means well. It can lead to poor inference results. We can obtain a fixed point algorithm to evaluate the means in exponential families for the approximating distribution. However, this does not solve the problem of underestimating the variances. In this paper, we propose a modified method of VBA with exponential families to first estimate the posterior mean and then improve the estimation of the posterior covariance. We demonstrate the performance of the procedure with an example.

Keywords: Variational Bayesian Approximation (VBA); Kullback–Leibler divergence; Mean–Field Variational Bayesian Approximation (MFVBA); Linear Response Variational Bayes (LRVB)

check for updates

Citation: Fallah Mortezanejad, S.A.; Mohammad-Djafari, A. Variational Bayesian Approximation (VBA) with Exponential Families and Covariance Estimation. *Phys. Sci. Forum* **2023**, *9*, 12. https://doi.org/10.3390/psf2023009012

Academic Editors: Udo von Toussaint and Roland Preuss

Published: 30 November 2023

1. Introduction

Bayesian computation uses Bayes' theorem to revise the probabilities of unknowns based on prior knowledge and data. When the likelihood expressions and prior are available, we can get the expression of the posterior law. However, this expression needs the denominator integral computation of the Bayes formula, called evidence. Approximate Bayesian Computation (ABC) are a class of algorithms used to perform Bayesian computation without exactly computing the evidence. The first class of these methods generate samples from the posterior law. We can mention all of the Monte Carlo sampling methods, such as slice sampling, nested sampling, and Markov chain Monte Carlo (MCMC) methods. The second category of these methods includes those that directly compute the means and variances, such as the Variational Bayesian Approximation (VBA) methods.

MCMC is a class of algorithms for sampling from a probability distribution and estimating the posterior distribution. Metropolis et al. (1953) [1] introduced the first MCMC algorithm called the Metropolis algorithm. Hastings (1970) [2] brought it closer to applied statistics, generalizing the Metropolis algorithm for asymmetric proposal distributions.

VBA methods were introduced by Jordan et al. (1998) [3]. They were motivated by the demand to improve the Bayesian inference for large and complex models intractable by conventional methods such as MCMC. VBA is a fast and robust scheme to perform Bayesian computations in high–dimensional problems. The main idea is to approximate the high–complexity posterior probability law with a simple and lower–complexity probability law using the Kullback–Leibler divergence as the approximation criterion.

In many cases, the VBA methods are faster than the MCMC methods. VBA can release two types of posterior approximations. The first is the free–form product of distributions such as the conjugated exponential family known as the Mean–Field Approximation (MFA). The second one is the fixed–form for posterior distributions like multivariate Gaussian with a proper parametrization for the model, Sarkka and Nummenmaa (2009) [4]. MFA is a general approach in the graphical models, particularly for the hierarchical Bayesian models.

The foundation of VBA is minimizing the Kullback–Leibler divergence (KL) between the approximated probability distribution and the exact posterior distribution. Rohde and Wand (2016) [5] worked on semi–parametric Mean–Field Variational Bayesian Approximation (MFVBA) as a combination of the KL measure and MFA. A significant drawback of MFVBA is that it underestimates the variable uncertainties and is uninformed concerning the variable covariance of models. Giordano et al. (2015, 2018) [6,7] rectified the MFVBA and named Linear Response Variational Bayes (LRVB) as being able to provide proper uncertainties.

MFVBA is a valuable approximation method for estimating the posterior mean in the Bayesian framework. While using an exponential family for approximation, we get a fixed point algorithm for reckoning the means. In this paper, we show how to use VBA with exponential families to approximate the posterior means and simultaneously evaluate the posterior covariance more precisely. Via an example, we also compare the MFVBA and MCMC posterior distributions.

This paper's organization is as follows: In Section 2, we demonstrate the main idea of MFVBA with a particular case in exponential family (EF) distributions. In Section 3, we modify the covariance matrix resulting from the MFVBA to get a better approximation. In Section 4, we give details of the covariance matrix expression in EF. We show that the covariance computation needs high–dimensional matrix inversion, which is very costly in high–dimensional problems. In Section 5, we provide an example of how to compute the posterior distribution and compare the computational costs of the proposed method and the MCMC. In Section 6, we present the main conclusion on the paper content.

2. Mean–Field Variational Bayesian Approximation (MFVBA)

VBA approximates posterior distributions when the exact expression of the posterior is too complex or costly. MFA simplifies the VBA by assuming that the final distribution is factorized into independent margins, which means that each unknown parameter has its distribution independent of the others.

Suppose that p is approximated by q^* and $\boldsymbol{\theta}$ is the unknown vector of parameters. We define $\boldsymbol{V} = \mathrm{Cov}_{q^*}(\boldsymbol{\theta})$ and $\boldsymbol{\Sigma} = \mathrm{Cov}_p(\boldsymbol{\theta})$. We know that $\boldsymbol{V} < \boldsymbol{\Sigma}$. $\boldsymbol{V}$ is a poor estimate (underestimate) of $\boldsymbol{\Sigma}$ even if $\boldsymbol{m}^* = \mathrm{E}_{q^*}\{\boldsymbol{\theta}\} = \mathrm{E}_p\{\boldsymbol{\theta}\}$. The question is how to find a better estimate for $\boldsymbol{\Sigma}$. One solution for $\boldsymbol{\Sigma}$ in the case of the exponential family, proposed by Giordano et al. (2015) [6], is $\boldsymbol{\Sigma} = (\boldsymbol{I} - \boldsymbol{V}\boldsymbol{H})^{-1}\boldsymbol{V}$ via the LRVB method, where $\boldsymbol{I}$ and $\boldsymbol{H}$ are an identity matrix and Hessian matrix of the log probability $\ln p$, respectively. To show this, let us go through the details step by step.

2.1. Main Idea

- General Bayesian framework: Given the likelihood $p(\mathcal{D}|\boldsymbol{\theta})$ and the prior $p(\boldsymbol{\theta})$, the expression of the posterior $p(\boldsymbol{\theta}|\mathcal{D})$ is given by:

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{D})}. \tag{1}$$

- MFVBA: Approximate $p(\boldsymbol{\theta}|x)$ by a separable $q(\boldsymbol{\theta})$:

$$q(\boldsymbol{\theta}) = \prod_j q_j(\theta_j), \tag{2}$$

by minimizing:

$$\mathrm{KL}[q:p] = \int q \ln \frac{q}{p} = \mathrm{E}_q\{\ln q\} - \mathrm{E}_q\{\ln p\}. \tag{3}$$

- Noting by $L = \mathrm{E}_q\{\ln p\}$ the expected posterior likelihood and by $S = -\mathrm{E}_q\{\ln q\}$ the entropy of q, we can also write:

$$q^* = \arg\min_q\{\mathrm{KL}[q:p]\} = \arg\max_q\{E = S + L\}. \tag{4}$$

- E is the evidence lower bound.

2.2. VBA and Exponential Family

- If q is chosen to be in an exponential family:

$$q(\boldsymbol{\theta}|\boldsymbol{\eta}) = \exp\left[\boldsymbol{\eta}'\boldsymbol{\theta} - A(\boldsymbol{\eta})\right], \tag{5}$$

then it is entirely characterized by its mean $\boldsymbol{m} = \mathrm{E}_q\{\boldsymbol{\theta}\}$ and so is:

$$q^*(\boldsymbol{\theta}|\boldsymbol{\eta}^*) = \exp\left[\boldsymbol{\eta}^{*\prime}\boldsymbol{\theta} - A(\boldsymbol{\eta}^*)\right], \tag{6}$$

then the same characterization by the mean $\boldsymbol{m}^* = \mathrm{E}_{q^*}\{\boldsymbol{\theta}\}$.
- We can then define the objective E as a function of $\boldsymbol{m}$, and the first–order condition of the optimality is:

$$\frac{\partial E}{\partial \boldsymbol{m}}\Big|_{\boldsymbol{m}=\boldsymbol{m}^*} = 0. \tag{7}$$

2.3. MFVBA, Exponential Family (EF) and Fixed Point Algorithm

- VBA+exponential family with $\mathrm{E}_q\{\boldsymbol{\theta}\}$:

$$\frac{\partial E}{\partial \boldsymbol{m}}\Big|_{\boldsymbol{m}=\boldsymbol{m}^*} = 0 \rightarrow \frac{\partial E}{\partial \boldsymbol{m}}\Big|_{\boldsymbol{m}=\boldsymbol{m}^*} + \boldsymbol{m} = \boldsymbol{m} \rightarrow M(\boldsymbol{m}) = \boldsymbol{m}. \tag{8}$$

- Iterating on this fixed point algorithm:

$$M(\boldsymbol{m}) = \boldsymbol{m}^{(k-1)} \quad \text{with} \quad \boldsymbol{M}(\boldsymbol{m}) := \frac{\partial E}{\partial \boldsymbol{m}} + \boldsymbol{m}, \tag{9}$$

converges to $\boldsymbol{m}^* = \mathrm{E}_{q^*}\{\boldsymbol{\theta}\}$.

3. VBA, EF and Covariance Estimation

- Noting from $\boldsymbol{V} = \mathrm{Cov}_{q^*}(\boldsymbol{\theta})$ and $\boldsymbol{\Sigma} = \mathrm{Cov}_p(\boldsymbol{\theta})$, we know that $\boldsymbol{V} < \boldsymbol{\Sigma}$.
- $\boldsymbol{V}$ is a poor estimate (underestimate) of $\boldsymbol{\Sigma}$ even if $\boldsymbol{m}^* = \mathrm{E}_{q^*}\{\boldsymbol{\theta}\} = \mathrm{E}_p\{\boldsymbol{\theta}\}$.
- The question is now: How to find a better estimate for $\boldsymbol{\Sigma}$.
- One solution proposed by Giordano et al. (2015) [6] is the LRVB.
The main idea is to perturb $p(\boldsymbol{\theta}|x)$ by some exponential family to obtain $p_{\boldsymbol{t}}(\boldsymbol{\theta}|x)$:

$$p_{\boldsymbol{t}}(\boldsymbol{\theta}|x) = p(\boldsymbol{\theta}|x)\exp\left[\boldsymbol{t}'\boldsymbol{\theta} + C(\boldsymbol{t})\right] \tag{10}$$

such that:

$$\ln p_{\boldsymbol{t}}(\boldsymbol{\theta}|x) = \ln p(\boldsymbol{\theta}|x) + \boldsymbol{t}'\boldsymbol{\theta} + C(\boldsymbol{t}) \tag{11}$$

- $C(\boldsymbol{t})$ is the cumulate generating function of $p_{\boldsymbol{t}}(\boldsymbol{\theta}|x)$, so

$$\boldsymbol{\Sigma} = \mathrm{Cov}_p(\boldsymbol{\theta}) = \frac{\partial^2 C(\boldsymbol{t})}{\partial \boldsymbol{t}'\partial \boldsymbol{t}}\Bigg|_{\boldsymbol{t}=\boldsymbol{0}} = \frac{\partial \mathrm{E}_{p_{\boldsymbol{t}}}\{\boldsymbol{\theta}\}}{\partial \boldsymbol{t}'}\Bigg|_{\boldsymbol{t}=\boldsymbol{0}} = \frac{\partial \boldsymbol{m}^*_{\boldsymbol{t}}}{\partial \boldsymbol{t}'}\Bigg|_{\boldsymbol{t}=\boldsymbol{0}} \tag{12}$$

4. Linear Perturbation Method for Better Covariance Estimation

To summarize this method:

- Perturb $p(\boldsymbol{\theta}|x)$ by an exponential family term to obtain:

$$p_{\boldsymbol{t}}(\boldsymbol{\theta}|x) = p(\boldsymbol{\theta}|x)\exp\left[\boldsymbol{t}'\boldsymbol{\theta} + C(\boldsymbol{t})\right].$$

Thus, we have:

$$\begin{aligned} -\mathrm{KL}[q:p_{\boldsymbol{t}}] &= -\,\mathrm{KL}[q:p] + \boldsymbol{t}'\boldsymbol{m} \longrightarrow E_{\boldsymbol{t}} = E + \boldsymbol{t}'\boldsymbol{m}, \\ \frac{\partial E}{\partial \boldsymbol{m}} &=0 \longrightarrow \boldsymbol{m} = \mathrm{E}_q\{\boldsymbol{m}\}, \quad M(\boldsymbol{m}) = \boldsymbol{m}, \end{aligned}$$

and finally:

$$E_{\boldsymbol{t}} = E + \boldsymbol{t}'\boldsymbol{m} \longrightarrow M_{\boldsymbol{t}}(\boldsymbol{m}) = M(\boldsymbol{m}) + \boldsymbol{t}. \tag{13}$$

- Replacing these relations, we have:

$$\frac{\partial \boldsymbol{m}^*_{\boldsymbol{t}}}{\partial \boldsymbol{t}'} = \left.\frac{\partial M_{\boldsymbol{t}}}{\partial \boldsymbol{m}'}\right|_{\boldsymbol{m}=\boldsymbol{m}^*_{\boldsymbol{t}}} \frac{\partial \boldsymbol{m}}{\partial \boldsymbol{t}} + \frac{\partial M_{\boldsymbol{t}}}{\partial \boldsymbol{t}} = \left.\frac{\partial M_{\boldsymbol{t}}}{\partial \boldsymbol{m}'}\right|_{\boldsymbol{m}=\boldsymbol{m}^*_{\boldsymbol{t}}} \frac{\partial \boldsymbol{m}}{\partial \boldsymbol{t}} + \boldsymbol{I}.$$

We obtain:

$$\boldsymbol{\Sigma} = \frac{\partial \boldsymbol{m}^*_{\boldsymbol{t}}}{\partial \boldsymbol{t}'} = \frac{\partial M}{\partial \boldsymbol{m}}\boldsymbol{\Sigma} + \boldsymbol{I} = \left(\frac{\partial^2 E}{\partial \boldsymbol{m}\partial \boldsymbol{m}'} + \boldsymbol{I}\right) + \boldsymbol{I} \longrightarrow \boldsymbol{\Sigma} = -\left(\frac{\partial^2 E}{\partial \boldsymbol{m}\partial \boldsymbol{m}'}\right),$$

and finally:

$$E = L + S \longrightarrow \boldsymbol{\Sigma} = -\left(\frac{\partial^2 L}{\partial \boldsymbol{m}\partial \boldsymbol{m}'} + \frac{\partial^2 S}{\partial \boldsymbol{m}\partial \boldsymbol{m}'}\right). \tag{14}$$

- Noting that the entropy of an exponential family is:

$$S = \boldsymbol{\eta}'\boldsymbol{m} + A(\boldsymbol{\eta}) \longrightarrow \frac{\partial S}{\partial \boldsymbol{m}} = -\boldsymbol{\eta} \longrightarrow \frac{\partial^2 S}{\partial \boldsymbol{m}\partial \boldsymbol{m}'} = -\frac{\partial \boldsymbol{\eta}}{\partial \boldsymbol{m}} = \boldsymbol{V}^{-1}.$$

We get:

$$\boldsymbol{\Sigma} = -\left(\frac{\partial^2 L}{\partial \boldsymbol{m}\partial \boldsymbol{m}'} + \frac{\partial^2 S}{\partial \boldsymbol{m}\partial \boldsymbol{m}'}\right)^{-1} = -(\boldsymbol{H} - \boldsymbol{V}^{-1})^{-1}.$$

- Finally, using a matrix inversion lemma, we hold:

$$\boldsymbol{\Sigma} = (\boldsymbol{I} - \boldsymbol{V}\boldsymbol{H})^{-1}\boldsymbol{V}. \tag{15}$$

4.1. Dealing with Nuisance Parameters

- If we are only interested in some of the variables $\boldsymbol{\alpha}$, i.e., $\boldsymbol{\theta} = (\boldsymbol{\alpha}, \boldsymbol{z})$ where $\boldsymbol{z}$ are nuisance parameters that we are not interested in, then we can write:

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_\alpha & \boldsymbol{\Sigma}_{\alpha z} \\ \boldsymbol{\Sigma}_{z\alpha} & \boldsymbol{\Sigma}_z \end{bmatrix}.$$

- Looking for the partition of $\boldsymbol{\theta} = (\boldsymbol{\alpha}, \boldsymbol{z})$, we obtain:

$$\boldsymbol{\Sigma}_\alpha = (\boldsymbol{I}_\alpha - \boldsymbol{V}_\alpha\boldsymbol{H}_\alpha - \boldsymbol{V}_\alpha\boldsymbol{H}_{\alpha z}(\boldsymbol{I}_z - \boldsymbol{V}_z\boldsymbol{H}_z)^{-1}\boldsymbol{V}_z\boldsymbol{H}_{z\alpha})^{-1}\boldsymbol{V}_\alpha.$$

4.2. The Whole Algorithm

- Having chosen the prior $p(\boldsymbol{\theta})$ and the likelihood $p(\mathcal{D}|\boldsymbol{\theta})$, find the expression of the posterior law:

$$p(\boldsymbol{\theta}|\mathcal{D}) \propto p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}).$$

- Choose an exponential family $q(\boldsymbol{\theta})$ and find the expressions of:
$$L = \mathrm{E}_q\{\ln p\}, \quad S = \mathrm{E}_q\{\ln q\}, \quad \text{and} \quad E = L + S,$$
as a function of the expectation parameters $\boldsymbol{m} = \mathrm{E}_q\{\boldsymbol{\theta}\}$.
- Find the expression of $\boldsymbol{M} = \nabla_{\boldsymbol{m}} E + \boldsymbol{m}$ and write the following fix point update $\boldsymbol{M}(\boldsymbol{m}) = \boldsymbol{m}^{(k-1)}$ until convergence.
- When $\boldsymbol{m}^\star$ is obtained, then compute:
$$\boldsymbol{V} = \frac{\partial^2 S}{\partial \boldsymbol{m} \partial \boldsymbol{m}'}|_{\boldsymbol{m}^\star} \quad \text{and} \quad \boldsymbol{H} = \frac{\partial^2 L}{\partial \boldsymbol{m} \partial \boldsymbol{m}'}|_{\boldsymbol{m}^\star}.$$
- Compute $\boldsymbol{\Sigma} = (\boldsymbol{I} - \boldsymbol{V}\boldsymbol{H})^{-1}\boldsymbol{V}$.

5. Numerical Experimentations of Normal–Poisson Distribution

The Normal–Poisson example is a non–conjugacy generalized mix model. The observed data are from a Poisson distribution named y_n and a design vector x_n, for $n = 1, \ldots, N$. The considered model is in Figure 1:

$$y_n | z_n \sim \mathcal{P}(\exp(z_n)), \tag{16}$$

where
$$z_n | \beta, \tau \sim \mathcal{N}\left(\beta x_n, \tau^{-1}\right), \quad \beta \sim \mathcal{N}\left(\mathbf{0}, \sigma_\beta^2\right), \quad \textit{and } \tau \sim \Gamma(\alpha_\tau, \beta_\tau).$$

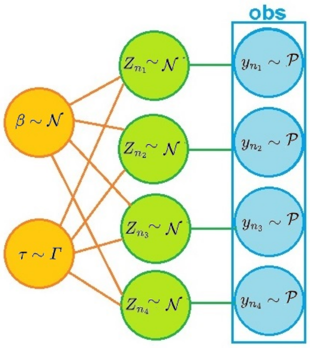

obs
$\beta \sim \mathcal{N}$
$\tau \sim \Gamma$
$Z_{n_1} \sim \mathcal{N}$
$Z_{n_2} \sim \mathcal{N}$
$Z_{n_3} \sim \mathcal{N}$
$Z_{n_4} \sim \mathcal{N}$
$y_{n_1} \sim \mathcal{P}$
$y_{n_2} \sim \mathcal{P}$
$y_{n_3} \sim \mathcal{P}$
$y_{n_4} \sim \mathcal{P}$

Figure 1. The Normal–Poisson hierarchical model pattern when $N = 4$ with sample size 100.

In the first step, we approximate the posterior joint distribution of β, τ and $\boldsymbol{z} = (z_1, \cdots, z_n)$ for $n = 1, \ldots, N$ via MFVBA:

$$q(\beta, \tau, \boldsymbol{z}) = q_{N+1}(\beta)\, q_{N+2}(\tau) \prod_{n=1}^{N} q_n(z_n).$$

We need to write $L = \ln p(\beta, \tau, \boldsymbol{z}, y)$:

$$\ln p(\beta, \tau, \boldsymbol{z}, y) = -\frac{\beta^2}{2\sigma_\beta^2} + (\alpha_\tau + \frac{N}{2} - 1)\ln(\tau) - \beta_\tau \tau - \frac{\tau}{2}\sum_{n=1}^{N} z_n^2 - \frac{\tau\beta^2}{2}\sum_{n=1}^{N} x_n^2 \tag{17}$$
$$+ \tau\beta \sum_{n=1}^{N} z_n x_n + \sum_{n=1}^{N} z_n y_n - \sum_{n=1}^{N} \exp(z_n) - \sum_{n=1}^{N} \ln(y_n!) + C,$$

where C is a constant and does not depend on unknown variables. We start with computing the distributions of z_n for $n = 1, \ldots, N$. For the simplicity of notations, we assume that n is a natural number in the interval $[1, N]$, and $q_{N+2\backslash n}$ means all $N + 2$ distribution items except the nth one, and $\langle \cdot \rangle_q$ means the expectation over density q. Thus, $q_n(z_n)$, the density of z_n, is equivalent to the following:

$$\langle \ln p(\beta, \tau, z, y) \rangle_{q_{N+2 \setminus n}} \propto -\frac{\langle \tau \rangle_{q_{N+2}}}{2} z_n^2 + (\langle \tau \rangle_{q_{N+2}} \langle \beta \rangle_{q_{N+1}} x_n + y_n) z_n - \exp(z_n).$$

Since this expression does not follow a specific distribution because of the existence of $\exp(z_n)$, we omit this part according to its expectation, which is a function of $\langle z_n \rangle_{q_n}$ and $\langle z_n^2 \rangle_{q_n}$. We can calculate the expectations using the Normal moment–generating distribution function. The logical reason is that its expectation depends only on $\langle z_n \rangle_{q_n}$ and $\langle z_n^2 \rangle_{q_n}$. Moreover, we have $\langle \tau \rangle_{q_{N+2}} = \frac{\alpha_\tau}{\beta_\tau}$ and $\langle \beta \rangle_{q_{N+1}} = 0$. Thus,

$$\langle \ln p(\beta, \tau, z, y) \rangle_{q_{N+2 \setminus n}} \propto -\frac{\alpha_\tau}{2\beta_\tau} \left(z_n^2 - \frac{2\beta_\tau y_n}{\alpha_\tau} z_n \right),$$

which means a Normal distribution for z_n, $\mathcal{N}(\frac{\beta_\tau \overline{y}_n}{\alpha_\tau}, \quad \frac{\beta_\tau}{\alpha_\tau})$ for $n = 1, \ldots, N$. The process is quite similar for $q_{N+2}(\tau)$ and $q_{N+1}(\beta)$. The corresponding equivalents of distributions for q_{N+2} and q_{N+1} are shown below, respectively:

$$\begin{aligned} \langle \ln p(\beta, \tau, z, y) \rangle_{q_{1,\ldots,N+1}} \propto \quad & (\alpha_\tau + \frac{N}{2} - 1) \ln(\tau) \\ & - \frac{\tau}{2} \left(2\beta_\tau + \sum_{n=1}^{N} (\langle z_n^2 \rangle_{q_n} + \langle \beta^2 \rangle_{q_{N+1}} x_n^2 - 2 \langle \beta \rangle_{q_{N+1}} x_n \langle z_n \rangle_{q_n}) \right), \end{aligned}$$

and

$$\langle \ln p(\beta, \tau, z, y) \rangle_{q_{1,\ldots,N,N+2}} \propto -\frac{\beta^2}{2\sigma_\beta^2} - \frac{\langle \tau \rangle_{q_{N+2}} \beta^2}{2} \sum_{n=1}^{N} x_n^2 + \langle \tau \rangle_{q_{N+2}} \beta \sum_{n=1}^{N} \langle z_n \rangle_{q_n} x_n.$$

Thus, we have:

$$\tau \sim \Gamma \left(\alpha_\tau + \frac{N}{2}, \quad \frac{1}{2} \left(2\beta_\tau + \frac{\beta_\tau}{\alpha_\tau} N + \frac{\beta_\tau^2}{\alpha_\tau^2} \sum_{n=1}^{N} \overline{y}_n^2 + \sigma_\beta^2 \sum_{n=1}^{N} x_n^2 \right) \right),$$

and

$$\begin{aligned} \beta \sim \mathcal{N} \Bigg(& \frac{\sigma_\beta^2 (2\alpha_\tau + N) \frac{\beta_\tau}{\alpha_\tau} \sum_{n=1}^{N} \overline{y}_n x_n}{2\beta_\tau + \frac{\beta_\tau}{\alpha_\tau} N + \frac{\beta_\tau^2}{\alpha_\tau^2} \sum_{n=1}^{N} \overline{y}_n^2 + \sigma_\beta^2 (2\alpha_\tau + N + 1) \sum_{n=1}^{N} x_n^2}, \\ & \frac{\sigma_\beta^2 \left(2\beta_\tau + \frac{\beta_\tau}{\alpha_\tau} N + \frac{\beta_\tau^2}{\alpha_\tau^2} \sum_{n=1}^{N} \overline{y}_n^2 + \sigma_\beta^2 \sum_{n=1}^{N} x_n^2 \right)}{2\beta_\tau + \frac{\beta_\tau}{\alpha_\tau} N + \frac{\beta_\tau^2}{\alpha_\tau^2} \sum_{n=1}^{N} \overline{y}_n^2 + \sigma_\beta^2 (2\alpha_\tau + N + 1) \sum_{n=1}^{N} x_n^2} \Bigg). \end{aligned}$$

In the second step, we calculate the variance diagonal matrix $\boldsymbol{V}$:

$$\boldsymbol{V} = \begin{bmatrix} \Sigma_\beta & 0 & 0 \\ 0 & \Sigma_\tau & 0 \\ 0 & 0 & \boldsymbol{\Sigma_z} \end{bmatrix},$$

where

$$\begin{aligned}\Sigma_\beta =& \frac{\sigma_\beta^2\Big(2\beta_\tau + \frac{\beta_\tau}{\alpha_\tau}N + \frac{\beta_\tau^2}{\alpha_\tau^2}\sum_{n=1}^N \bar{y}_n^2 + \sigma_\beta^2 \sum_{n=1}^N x_n^2\Big)}{2\beta_\tau + \frac{\beta_\tau}{\alpha_\tau}N + \frac{\beta_\tau^2}{\alpha_\tau^2}\sum_{n=1}^N \bar{y}_n^2 + \sigma_\beta^2(2\alpha_\tau + N + 1)\sum_{n=1}^N x_n^2},\\ \Sigma_\tau =& \frac{2(2\alpha_\tau + N)}{\Big(2\beta_\tau + \frac{\beta_\tau}{\alpha_\tau}N + \frac{\beta_\tau^2}{\alpha_\tau^2}\sum_{n=1}^N \bar{y}_n^2 + \sigma_\beta^2 \sum_{n=1}^N x_n^2\Big)^2},\end{aligned}$$

and

$$\Sigma_{z_n} = \frac{\beta_\tau}{\alpha_\tau}, \quad n = 1, \ldots, N.$$

Thus, $\boldsymbol{\Sigma_z} = diag(\Sigma_{z_1}, \cdots, \Sigma_{z_N})$. The third step is to compute the Hessian matrix $\boldsymbol{H}$:

$$\boldsymbol{H} = \begin{bmatrix} \frac{\partial^2 L}{\partial \beta^2} & \frac{\partial^2 L}{\partial \beta \partial \tau} & \frac{\partial^2 L}{\partial \beta \partial \boldsymbol{z}} \\ \frac{\partial^2 L}{\partial \tau \partial \beta} & \frac{\partial^2 L}{\partial \tau^2} & \frac{\partial^2 L}{\partial \tau \partial \boldsymbol{z}} \\ \frac{\partial^2 L}{\partial \boldsymbol{z} \partial \beta} & \frac{\partial^2 L}{\partial \boldsymbol{z} \partial \tau} & \frac{\partial^2 L}{\partial \boldsymbol{z}^2} \end{bmatrix}.$$

Since $\boldsymbol{H}$ is symmetric in our case study, we need to calculate the main diagonal and the upper triangle. This covariance matrix depends on the values of $\boldsymbol{m}$:

$$\boldsymbol{m} = \Big(E_q(\beta), E_q(\beta^2), E_q(\tau), E_q(\ln(\tau)), E_q(z_1), E_q(z_1^2), \ldots, E_q(z_N), E_q(z_N^2)\Big)^T.$$

Thus, the diagonal of $\boldsymbol{H}$ as a function of $\boldsymbol{m}$ is shown below:

$$\begin{aligned}\frac{\partial^2 L}{\partial \beta^2} =& -\frac{1}{\sigma_\beta^2} - E_q(\tau)\sum_{n=1}^N x_n^2,\\ \frac{\partial^2 L}{\partial \tau^2} =& -(\alpha_\tau + \frac{N}{2} - 1)E_q\Big(\frac{1}{\tau^2}\Big),\\ \frac{\partial^2 L}{\partial z_n^2} =& -E_q(\tau) - E_q(\exp(z_n)), \quad n = 1, \ldots, N,\end{aligned}$$

where L is defined in Equation (17). The upper triangle is:

$$\begin{aligned}\frac{\partial^2 L}{\partial \beta \partial \tau} =& -E_q(\beta)\sum_{n=1}^N x_n^2 + \sum_{n=1}^N E_q(z_n)x_n,\\ \frac{\partial^2 L}{\partial \beta \partial z_n} =& E_q(\tau)x_n, \quad n = 1, \ldots, N,\\ \frac{\partial^2 L}{\partial \tau \partial z_n} =& -E_q(z_n) + E_q(\beta)x_n, \quad n = 1, \ldots, N.\end{aligned}$$

The last step is to compute $\boldsymbol{\Sigma}$ using (15). For numerical experimentation, we generate data from the Normal–Poisson model with these descriptions:

$$\sigma_\beta^2 = 10,\ \alpha_\tau = 1,\ \beta_\tau = 1,\ N = 4.$$

We generate x_n for $n = 1, \ldots, N$ from the Normal distribution $\mathcal{N}(2, 1)$. The number of the reputation for vector y_n is 100, so y_n is a matrix $N \times 100$. The only data we use are the observation matrix y_n and vector x_n. The covariance matrix $\boldsymbol{V}$ from MFVBA is a diagonal matrix:

$$\boldsymbol{V} = \begin{bmatrix} 1.48927 & 0 & \mathbf{0}^T \\ 0 & 0.00071 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} & \boldsymbol{I} \end{bmatrix},$$

where $\mathbf{0}$ and $\boldsymbol{I}$ are the zero vector with N dimensions and the identity matrix with $N \times N$ dimensions, respectively. The Hessian matrix $\boldsymbol{H}$ and matrix $\boldsymbol{I} - \boldsymbol{V}\boldsymbol{H}$ are:

$$\boldsymbol{H} = \begin{bmatrix} -0.67147 & 0.09617 & 0.05557 & 0.03031 & 0.13360 & 0.07026 \\ 0.09617 & -1639.9471 & -0.21738 & -0.02130 & 0.02651 & 0.06653 \\ 0.05557 & -0.21738 & -2.20665 & 0 & 0 & 0 \\ 0.03031 & -0.02130 & 0 & -1.78014 & 0 & 0 \\ 0.13360 & 0.02651 & 0 & 0 & -1.86901 & 0 \\ 0.07026 & 0.06653 & 0 & 0 & 0 & -1.69561 \end{bmatrix},$$

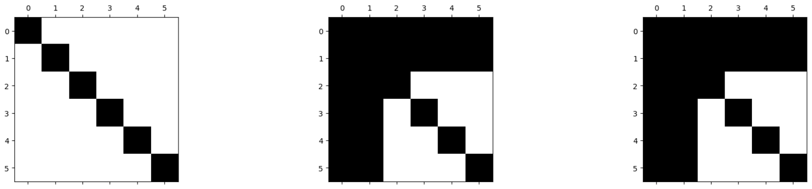
and

$$\boldsymbol{I} - \boldsymbol{V}\boldsymbol{H} = \begin{bmatrix} 2 & -0.14322 & -0.08276 & -0.04514 & -0.19897 & -0.10464 \\ -6.8318 \times 10^{-5} & 2.16501 & 0.00015 & 1.5130 \times 10^{-5} & -1.8832 \times 10^{-5} & -4.7264 \times 10^{-5} \\ -0.05557 & 0.21738 & 3.20665 & 0 & 0 & 0 \\ -0.03031 & 0.02130 & 0 & 2.78014 & 0 & 0 \\ -0.13360 & -0.02651 & 0 & 0 & 2.86901 & 0 \\ -0.07026 & -0.06653 & 0 & 0 & 0 & 2.69561 \end{bmatrix}.$$

The covariance matrix via LRVBA is:

$$\boldsymbol{\Sigma} = \begin{bmatrix} 0.74986 & 2.3409 \times 10^{-5} & 0.01299 & 0.00818 & 0.03492 & 0.01955 \\ 2.3408 \times 10^{-5} & 0.00033 & -2.1838 \times 10^{-5} & -2.2586 \times 10^{-6} & 4.1220 \times 10^{-6} & 8.7090 \times 10^{-6} \\ 0.01299 & -2.1838 \times 10^{-5} & 0.31208 & 0.00014 & 0.00060 & 0.00034 \\ 0.00817 & -2.2586 \times 10^{-6} & 0.00014 & 0.35978 & 0.00038 & 0.00021 \\ 0.03492 & 4.1220 \times 10^{-6} & 0.00060 & 0.00038 & 0.35018 & 0.00091 \\ 0.01955 & 8.7090 \times 10^{-6} & 0.00034 & 0.00021 & 0.00091 & 0.37148 \end{bmatrix}. \quad (18)$$

The sparsity patterns for the $\boldsymbol{V}$, $\boldsymbol{H}$ and $\boldsymbol{I} - \boldsymbol{V}\boldsymbol{H}$ matrixes are shown in Figure 2.



(**a**) (**b**) (**c**)

Figure 2. Sparsity patterns for the matrices using model (16). (**a**) VBA covariance matrix $\boldsymbol{V}$; (**b**) Hessian matrix $\boldsymbol{H}$; (**c**) $\boldsymbol{I} - \boldsymbol{V}\boldsymbol{H}$ matrix.

In addition, we simulate the posterior distribution via MCMC. This simulation has five chains and one divergence. The estimated marginal distribution for each unknown variable is shown in Figures 3 and 4 via MFVBA and MCMC, respectively. More details are in Table 1. The joint distribution of unknown variables is approximated via MFVBA and MCMC. The MFVBA distribution is the marginal density factorial of the unknown variables. Therefore, the variables are independent in MFVBA, and the covariance matrix is diagonal. LRVBA is a correction method on the covariance matrix. Thus, the matrix is no longer diagonal as presented in (18). Also, we add more details about the MCMC results in Table 1. The Monte Carlo Standard Error (MCSE) is a chain–accurate measure and provides a quantitative suggestion of how big the estimation noise is. The noises are small based on the mean and sd of the MCSE. The ess–bulk column delivers an evaluated bulk influential sample size using rank–normalised draws. The ess–tail column creates an approximated tail practical sample size by computing the effective minimum sample sizes between the 5% and 95% quantiles. The last column is the r–hat convergence diagnostic, which compares the between and within–chain estimation for model variables and other univariate quantities.

Table 1. Summary of the margin densities for unknown variables.

Unknown	MFVBA Method		LRVBA Method	MCMC Method								
	mean	**sd**	**sd**	**mean**	**sd**	**hdi–3%**	**hdi–97%**	**MCSE–mean**	**MCSE–sd**	**ess–bulk**	**ess–tail**	**r–hat**
β	0.044	1.220	0.866	-1.616	1.769	-5.175	1.620	0.148	0.194	186.0	43.0	1.11
τ	0.047	0.027	0.018	0.361	0.188	0.057	0.685	0.024	0.017	38.0	51.0	1.11
z_0	0.270	1	0.559	-1.316	0.177	-1.664	-1.028	0.008	0.006	468.0	316.0	1.02
z_1	0.050	1	0.600	-3.051	0.410	-3.783	-2.252	0.021	0.015	382.0	372.0	1.00
z_2	0.100	1	0.592	-2.375	0.332	-3.029	-1.790	0.015	0.011	476.0	415.0	1.02
z_3	0	1	0.610	-7.334	4.641	-14.366	-3.311	0.868	0.620	45.0	55.0	1.13

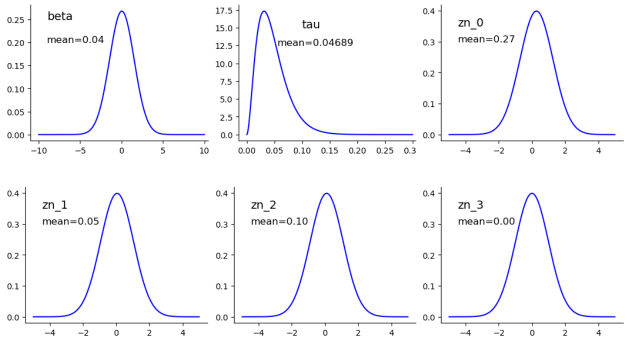


Figure 3. The final estimations of margin densities for each variable via MFVBA method.

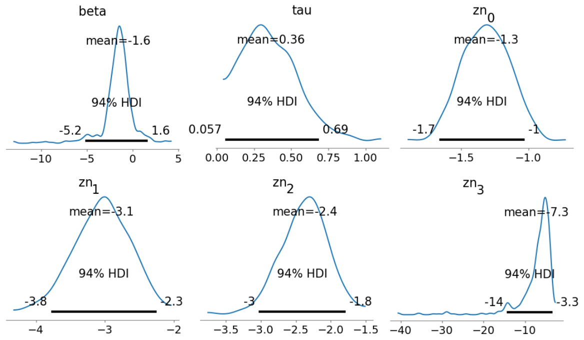


Figure 4. The final margins via MCMC method.

6. Conclusions

Bayesian computations are used to infer unknown parameters from data. ABC methods are applied to perform approximate computation, particularly in high–dimensional problems. Two classes of approximate computational methods are MCMC and VBA. MCMC generates samples from the probability function to approximate the posterior distribution. It generates a sequence of samples that converges to the target distribution. The VBA approximates the expression of the joint posterior distribution of unknown parameters in a complex model using a simple expression. VBA has some advantages over MCMC, such as being faster, more scalable and versatile. It also guarantees convergence to a local minimum of KL divergence.

VBA's drawback is that it underestimates the covariance matrix of the unknowns. For this aim, we use LRVB to approximate the covariance more precisely. However, the LRVB computations still need the inversion of some matrixes, which is costly when the unknown parameters' dimensions are high. We work on a numerical example to show the whole process of VBA and LRVB as well as the performance and ability of the proposed method and compare it with the classical MCMC results. As a result of the numerical example, the VBA implementation was much shorter than the MCMC, and we had an explicit form of joint distribution with a modified covariance matrix with the LRVB aid.

Author Contributions: Conceptualization, S.A.F.M. and A.M.-D.; methodology, S.A.F.M. and A.M.-D.; software, S.A.F.M.; validation, A.M.-D.; formal analysis, S.A.F.M. and A.M.-D.; investigation, S.A.F.M. and A.M.-D.; resources, S.A.F.M. and A.M.-D.; data curation, S.A.F.M. and A.M.-D.; writing—original draft preparation, S.A.F.M. and A.M.-D.; writing—review and editing, S.A.F.M. and A.M.-D.; visualization, S.A.F.M. and A.M.-D.; supervision, A.M.-D.; project administration, A.M.-D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing is not applicable to this article.

Conflicts of Interest: The author Ali Mohammad-Djafari is Scientific Research Leader of Shanfeng Company. The authors declare no conflict of interest.

References

1. Metropolis, N.; Rosenbluth, A.W.; Rosenbluth, M.N.; Teller, A.H.; Teller, E. Equation of state calculations by fast computing machines. *J. Chem. Phys.* **1953**, *21*, 1087–1092. [CrossRef]
2. Hastings, W.K. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* **1970**, *57*, 97–109. [CrossRef]
3. Jordan, M.I.; Ghahramani, Z.; Jaakkola, T.S.; Saul, L.K. An introduction to variational methods for graphical models. In *Learning in Graphical Models*; Springer: Dordrecht, The Netherlands, 1998; pp. 105–161.
4. Sarkka, S.; Nummenmaa, A. Recursive noise adaptive Kalman filtering by variational Bayesian approximations. *IEEE Trans. Autom. Control* **2009**, *54*, 596–600. [CrossRef]
5. Rohde, D.; Wand, M.P. Semiparametric mean field variational Bayes: General principles and numerical issues. *J. Mach. Learn. Res.* **2016**, *17*, 5975–6021.
6. Giordano, R.J.; Broderick, T.; Jordan, M.I. Linear response methods for accurate covariance estimates from mean field variational Bayes. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 1441–1449.
7. Giordano, R.; Broderick, T.; Jordan, M.I. Covariances, robustness and variational bayes. *J. Mach. Learn. Res.* **2018**, *19*, 1–49.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.